Computers & Operations Research 54 (2015) 1–11

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Minimizing weighted earliness–tardiness on parallel machines using hybrid metaheuristics

R. Alvarez-Valdes [a,*], J.M. Tamarit [a], F. Villa [b]

[a] University of Valencia, Department of Statistics and Operations Research, Burjassot, Valencia, Spain
[b] Polytechnic University of Valencia, Department of Applied Statistics and Operations Research, and Quality, Valencia, Spain

ARTICLE INFO

Available online 6 September 2014

Keywords:
Scheduling
Earliness-tardiness
Parallel machines
Hybrid metaheuristics
Path Relinking
Scatter Search

ABSTRACT

We consider the problem of scheduling a set of jobs on a set of identical parallel machines where the objective is to minimize the total weighted earliness and tardiness penalties with respect to a common due date. We propose a hybrid heuristic algorithm for constructing good solutions, combining priority rules for assigning jobs to machines and a local search with exact procedures for solving the one-machine subproblems. These solutions are then used in two metaheuristic frameworks, Path Relinking and Scatter Search, to obtain high quality solutions for the problem.

The algorithms are tested on a large number of test instances to assess the efficiency of the proposed strategies.

The results show that our algorithms consistently outperform the best reported results for this problem.

1. Introduction

In Just-In-Time scheduling we are concerned not only with job tardiness but also with earliness. Tardy jobs, completed after their due date, have negative effects such as contract penalties, customer discontent, loss of sales and loss of reputation, but early jobs, completed before their due date, also have non-desirable effects such as inventory carrying costs, storage and insurance costs, and product deterioration. Therefore, criteria involving both earliness and tardiness costs are receiving increased attention in machine scheduling research (see the books by Józefowska [1] and by Rios-Mercado and Rios-Solis [2] for surveys of models and algorithms in this area).

In this paper we consider the problem of scheduling a set of jobs on identical parallel machines where the objective is to minimize the total weighted earliness and tardiness penalties with respect to a common due date. In practice, problems with a common due date appear when a set of products has to be sent together to a client or when a set of components is produced to be assembled in a later phase.

The problem can be defined as follows. There are n jobs to be processed on M identical parallel machines. All the jobs have the same due date d. For each job i, its processing time p_i, and its penalties per period of earliness α_i and per period of tardiness β_i, are known. No preemption is allowed, all the jobs are available at time zero and the machine is continuously available for work. If we denote the completion time of job i by C_i, the objective is $min\sum_i^n \alpha_i E_i + \beta_i T_i$, where $E_i = max\{d - C_i, 0\}$ and $T_i = max\{C_i - d, 0\}$.

Therefore, there is a double problem to be solved: an assignment problem of the n jobs to the M machines and a sequencing problem on each machine to decide which jobs will finish before their due date, incurring an earliness penalty, which ones will be completed after their date, incurring a tardiness penalty, and if there will be a job finishing exactly on the due date, without any cost.

When tackling this objective function, two cases can be distinguished. We consider a problem as unrestricted, following the definition provided by Webster [3], if the optimal cost cannot decrease with increases in the common due date. In practice, this means that we can put as many jobs as we want before the due date because the interval $(0, d)$ is large enough. In the restrictive case, the due date affects the optimal schedule. In this paper we consider that we are dealing with the restrictive case and therefore we will have to check whether the jobs we want to process before the due date really do fit into the interval $(0, d)$. Obviously, the algorithms that solve the restrictive case also solve the unrestricted one.

According to the classification by Graham et al. [4], the problem can be denoted as $P|d^i = d^r|\sum_i(\alpha_i E_i + \beta_i T_i)$, where P is the identical parallel machine environment and $d^i = d^r$ means that all the jobs share the same restrictive due date. Several common due-date scheduling surveys have been done by Baker and Scudder [5], Gordon et al. [6] and Lauff and Werner [7]. The restrictive one-machine

* Corresponding author. Tel.: +34963544308; fax: +34963543238.
E-mail address: ramon.alvarez@uv.es (R. Alvarez-Valdes).

http://dx.doi.org/10.1016/j.cor.2014.08.020
0305-0548/© 2014 Elsevier Ltd. All rights reserved.

problem $1|d^i = d^r|\sum_i(\alpha_i E_i + \beta_i T_i)$ is already *NP-hard*, and the parallel case is not any easier. In fact, $P|d^i = d^r|\sum_i(\alpha_i E_i + \beta_i T_i)$ inherits the characteristic of being NP-hard from the parallel machine problem $P||\sum_i w_i C_i$ (see Brucker [8]).

There are not many studies on this problem. Sun and Wang [9] consider the identical parallel machine problem with a loose common due date $P|d^i = d^l|\sum_i(\alpha_i E_i + \beta_i T_i)$ (where d^l stands for a loose due date), where the weight w_i of each job is proportional to its processing time. They show that even for $M=2$, the problem is *NP-hard* in the ordinary sense. They formulate a dynamic programming algorithm and develop two list-scheduling heuristics. Chen and Powell [10] propose a column generation algorithm for the problem with a loose common due date $P|d^i = d^l|\sum_i(\alpha_i E_i + \beta_i T_i)$ and they solve instances of up to 60 jobs to optimality with a branch & bound algorithm. Rios-Solis and Sourd [11] are the only authors who have addressed the restrictive case $P|d^i = d^r|\sum_i(\alpha_i E_i + \beta_i T_i)$, developing a neighborhood search algorithm. The related problem in which each job has a different due date has been studied by Kedad-Sidhoum et al. [12], who propose lower bounds based on relaxations of time-indexed integer formulations and a simple and effective heuristic.

Any algorithm that solves this problem has two decisions to address: how to assign jobs to machines and how to sequence the jobs assigned to each machine. Our approach to these two questions, and therefore our contributions in this study, are as follows:

- We first develop several heuristic rules for assigning jobs to machines. We use two different strategies and design several priority rules based on combinations of job characteristics (processing times, earliness and tardiness penalties). We also include an assignment rule that learns from the other rules.
- The set of jobs assigned to each machine is then sequenced by solving the quadratic programming models we designed in an earlier study for the one-machine subproblems (see Alvarez-Valdes et al. [13]). Our previous results show that even for short computing times these models can provide very good, if not optimal, solutions to the sequencing problem.
- The two procedures described above are used not to build a solution, but a set of feasible solutions, one for each assignment criterion. We cannot expect that any of these criteria could provide a good job assignment for every possible instance, but we think that each criterion can contribute some good partial assignments which could be combined to obtain better solutions. In order to achieve this, we have designed two procedures: a Path Relinking algorithm, which builds paths in the solution space, linking the individual solutions provided by the first two phases, and a Scatter Search algorithm which builds new solutions by systematically combining each pair of solutions.
- These metaheuristic procedures obtain high quality solutions with reduced computing times even for large problems with 200 jobs, improving on the results reported in the literature. The comparison with the lower bound by Kedad-Sidhoum et al. [12] also shows that the solutions obtained are very near to optimal.

The remainder of the paper is organized as follows. In Section 2 we summarize the results of the one-machine models used in the algorithms. In Section 3 we describe the parallel machine problem in detail and outline the algorithmic scheme. Section 4 contains the constructive algorithms, while Section 5 is devoted to the local search. Sections 6 and 7 describe the Path Relinking and Scatter Search procedures, respectively. Section 8 contains the test instances and the computational results. Finally, in Section 9 we draw some conclusions and discuss future research.

2. The one-machine problem

In the one-machine problem there is a set of jobs to be processed on one machine, all of them with the same due date d. The one-machine problem has been extensively studied. From previous studies we know that there is always an optimal solution satisfying three properties:

1 An optimal schedule does not contain any idle time between consecutive jobs (Cheng and Kahlbacher [14]).
2 The optimal schedule is *V-shaped* around the common due date. Jobs completed before or on the due date are scheduled in non-increasing order of p_i/α_i, and jobs starting on or after the due date are scheduled in non-decreasing order of p_i/β_i (Baker and Scudder [5]).
3 In the optimal schedule, either the first job starts at time zero or there is a job finishing on the due date (Hoogeveen and van de Velde [15]).

According to Property 3, in a previous study (Alvarez-Valdes et al. [13]) we developed two different quadratic models: Model 1 for solving the instances where the optimal solution has a job finishing on the due date and Model 2 for those where the optimal solution starts at time zero.

The efficiency of both models was tested using the set of 280 test instances generated by Biskup and Feldman [16]. The set includes instances with 10, 20, 50, 100, 200, 500 and 1000 jobs, with different due-date tightness.

The main findings of that computational study were:

- Model 1 was solved very fast, even for large instances of 1000 jobs. Moreover, even if the solution process is truncated by a tight time limit, the solutions obtained were very near the optimal solutions. Therefore, solving this Model 1 with limited time can be seen as a good heuristic method for the one-machine earliness–tardiness problem.
- Model 2 was much slower to solve. For instances of more than 20 jobs there is no guarantee that a good solution can be found in a reasonable time.

In this study, when using Models 1 and 2 for solving one-machine subproblems, we adopt the following strategy:

- For subproblems involving up to 15 jobs, both Models are solved and the best solution is kept.
- For subproblems with more than 15 jobs, only Model 1 is solved.

3. Solving the parallel machine problem

For solving the parallel machine problem we could have tried the quadratic models developed in the previous section, adapting them to M machines. However, the resulting model is much more complex and it is not able to produce good solutions in reasonable times. Therefore, we propose an algorithmic scheme in which the one-machine models are used, but combined with heuristic priority rules and a local search.

The solution process is divided into three phases:

- *Phase 1: Constructing a set of feasible solutions*
  Phase 1 is in turn divided into two steps:
  1. Assign jobs to machines according to heuristic criteria
  2. Solve the corresponding one-machine subproblems using the models in the previous section